Drive2Local Project Documentation

Fabian Caraballo, Clark Chen, Joseph Shea-Bianco, Creston Wilson, Kaijing Zhang

February 18th, 2019

# Table of Contents

## [Project Plan](#_r967jnq6wwbs)

1. Purpose and Audience
2. Project Background
3. Team Roles and Responsibilities
4. Risks and Risk Mitigation
5. Process
6. Mechanisms, methods, techniques
7. Detailed schedule and milestones
8. Resources and References
9. Meeting Notes

## [Software Requirements Specification (SRS)](#_s14iv63veypw)

1. Introduction
2. Concept of Operations (from the User’s Point of View)
3. Behavioral Requirements
4. Developmental Quality Attributes
5. Fundamental Assumptions

## [Software Design Specification (SDS)](#_izgh4k8gxcu1)

1. Software Architecture
2. Module Interface Specification

## [Quality Assurance Plan](#_oltdb2zb8x0q)

1. Reviews
2. Testing

## [Software Documentation](#_ep0mbau8kxgr)

## [Developer Logs](#_7fogetsg5j1o)

# I. Project Plan

# Drive2Local

Revision History:

|  |  |  |
| --- | --- | --- |
| Date | Author | Description |
| 02/18/19 | Caraballo, Chen, Shea-Bianco, Wilson, Zhang | Initial draft |
| 02/21/19 | Wilson | Edit Process |
| 02/26/19 | Wilson | Edit Project Background |

## Purpose and Audience

This document is written for the instructor of CIS 422, Anthony Hornof, as well as the development team which consists of Fabian Caraballo, Clark Chen, Joseph Shea-Bianco, Creston Wilson, and Kaijing Zhang.

The purpose of this document is to inform its audience of the people involved, their roles in the process, and the project management decisions made by the development team.

## Project Background

Version control has became very popular within the last decade and according to a study from 2016, 78% of students use Google Drive for collaborative works[[1]](#footnote-0), as it allows real-time collaboration, version control, contribution tracking, and a relatively safe guarantee against file losses. Google Drive is a free cloud service that allows anyone to access documents from any machine as long as they have a Google account. As useful as it is, there are some aspects of Google Drive that we wish to improve upon. Some of the issues of Google Drive include data loss and difficulty recovering files promptly.

For this project, we are creating a program that allows anyone using Google Drive to backup all of their files locally. This will allow for greater data redundancy and users with access to files they may not have had access to otherwise due to permission changes or loss of an internet connection. Our software will also provide users with logging information, filters for backups, compression, and backup rotation.

There are other pieces of software that tries to sync or backup local files from cloud storage providers, specifically Backup and Sync by Google. We plan to differentiate our software by providing scheduled backups, rotation of backups, filtering of files to be backed up, and remove any syncing between local backups and Google Drive files.

## Team Roles and Responsibilities

|  |  |  |
| --- | --- | --- |
| Role | Team members taking on the role | Artifacts for which the role is responsible |
| Requirements Analyst (1) | Primary: Shea-Bianco  Backup: Zhang | Conops, Software Requirements Specification |
| Architect (1) | Primary: Wilson  Backup: Shea-Bianco | Software Design Specification |
| Developer (>1) | Primary: Caraballo, Chen, Shea-Bianco  Backup: Wilson, Zhang | Module implementation |
| Tester & Integrator (>1) | Primary: Wilson, Shea-Bianco  Backup: Caraballo, Chen, Zhang | Test planning, Module tests, System generation and verification plan, test results report |
| Project Manager (1) | Primary: Zhang  Backup: Chen | Communication, project plan, project measures, retrospective report |
| Configuration Control  (build master) | Primary: Shea-Bianco  Backup: Wilson | Configuration management policy (coordinate policy with team members, ensure build works). |

## 4. Risks and Risk Mitigation

|  |  |  |
| --- | --- | --- |
| Risk | Description | Mitigation |
| Merge Conflicts | When two people work on the same file and push their changes with Git, the files may not combine properly. | Work with the people pushing to ensure the combined file functions properly |
| Scheduling Conflicts | When team members are unable to coordinate due to prior commitments. | Communicate our schedules so that the team can continually plan. |
| Behind Schedule | Milestones are not met in time. Which results in pushing future milestones further out on the schedule. | Ensure that everyone communicates with each other about where they are at with their assigned tasks. If they need help, they ask for it. |

## 5. Process

Team members will meet at least once a week either in person or on Discord over voice chat. We will use Google Docs to collaborate on documentation. Github will be utilized as version control for the code needed to implement the project. In order to stay on schedule we will use a Gantt chart as a timeline. The team will use a group chat to keep in contact in between meetings.

We chose this process, because it worked for the previous project. We have assigned different roles to different team members to fit the project.

## 6. Mechanisms, Methods, techniques

Our team has elected to use Google Drive to contribute to and store our documentation. This is because Google Drive allows us to track contributions, permits simultaneous collaboration, and is accessible from anywhere. We are using GitHub to store our code for the version control and contribution tracking benefits it offers.

## 7. Detailed schedule and milestones

* Gantt Chart

## 8. Resources and References

Any questions relating to this or any related documentation or the project itself may be directed to any members of the development team at the following email addresses:

* Fabian Caraballo: [fpc@uoregon.edu](mailto:fpc@uoregon.edu)
* Clark Chen: [zchen@uoregon.edu](mailto:zchen@uoregon.edu)
* Joseph Shea-Bianco: [jsheabia@uoregon.edu](mailto:jsheabia@uoregon.edu)
* Creston Wilson: [crestonw@uoregon.edu](mailto:crestonw@uoregon.edu)
* Kaijing Zhang: [kaijingz@uoregon.edu](mailto:kaijingz@uoregon.edu)

## 

## 

## Meeting Notes

#### Meeting 1:

* **Date**: 02/18/19
* **Attendees**: All staff
* **Agenda**: Initial Project Documentation
* **Action items**:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Task and Deliverable | Assignee | Start Date | End Date | Estimated Time |
| Project Plan, SRS, SDS | Chen,  Shea-Bianco, Zhang | 02/18/19 | 2/21/19 | 180 mins |

#### Meeting 2:

* **Date**: 02/20/19
* **Attendees**: Chen, Shea-Bianco, Zhang
* **Agenda**: SRS/SDS/Quality Assurance Plan
* **Action items**:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Task and Deliverable | Assignee | Start Date | End Date | Estimated Time |
| SRS, SDS | Caraballo, Chen, Shea-Bianco, Wilson, Zhang | 2/21/19 | 2/21/19 | 180 mins |

#### 

#### Meeting 3:

* **Date**: 02/26/19
* **Attendees**: Chen, Shea-Bianco, Wilson
* **Agenda**: GUI/Update SRS/Create user diagram
* **Action items**:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Task and Deliverable | Assignee | Start Date | End Date | Estimated Time |
| SRS | Chen, Shea-Bianco, Wilson | 2/26/19 | 2/26/19 | 40 mins |
| GUI | Chen, Zhang | 2/26/19 | 3/05/19 | 6 days |

#### Meeting 4:

* **Date**: 02/27/19
* **Attendees**: All staff
* **Agenda**: Task Allocation
* **Action items**:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Task and Deliverable | Assignee | Start Date | End Date | Estimated Time |
| Object Definition Module | Caraballo, Shea-Bianco, | 2/27/19 | 03/03/19 | 4 days |
| Configuration Module | Chen,  Zhang | 2/27/19 | 03/03/19 | 4 days |
| API Access Module | Shea-Bianco, Wilson, Caraballo, Chen, Zhang | 2/27/19 | 03/03/19 | 4 days |

#### 

#### Meeting 5:

* **Date**: 03/11/19
* **Attendees**: All staff
* **Agenda**: Task Allocation, Finish coding, and documentation
* **Action items**:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Task and Deliverable | Assignee | Start Date | End Date | Estimated Time |
| System Testing |  | 3/11/19 | 03/13/19 | 4 days |
| Documentation Finalization |  | 3/11/19 | 03/13/19 | 4 days |
| Turn in project |  | 3/11/19 | 03/13/19 | 2 days |

# 

# II. Software Requirements Specification (SRS)

Revision History:

|  |  |  |
| --- | --- | --- |
| Date | Author | Description |
| 02/18/19 | Caraballo, Chen, Shea-Bianco, Wilson, Zhang | Initial draft |
| 02/20/19 | Chen, Shea-Bianco, Zhang | Edit and complete the initial draft |
| 02/26/2019 | Shea-Bianco, Wilson | Add stakeholder information and user diagram |
| 02/28/2019 | Shea-Bianco | Add additional software dependencies |

## Introduction

### 1.1. Intended Audience and Purpose

This document is written for the stakeholders of Drive2Local, primarily consisting of the target audience for Drive2Local. This target audience includes data-security-conscious end-users who wish to have backups of cloud data on their local devices. Examples of these users are college students and business professionals who keep work in Google Drive, but wish to maintain local backups for additional security.

Additional stakeholders include the instructor of CIS 422, Anthony Hornof, as well as the development team which consists of Fabian Caraballo, Clark Chen, Joseph Shea-Bianco, Creston Wilson, and Kaijing Zhang.

The purpose of this document is to inform its audience of the requirements of Drive2Local, in terms of operation, functionality, and procedure.

## Concept of Operations

### 2.1. System Context

Drive2Local requires a Google account and access to Google Drive, as well as sufficient storage on a local device to hold the backups created by Drive2Local.

Our system utilizes the Google Drive API to download files, then stores the resulting files on their local device.

### 2.2. System capabilities

After configuring filetype filters, timing, logging, rotation settings, and login credentials, Drive2Local will automatically download files of the types specified in the filetype filters from a Google Drive, compress the files, and store them, automatically removing backups older than a number of days specified in the rotation settings and submitting log information to a logging directory.

### 2.3. Use Cases

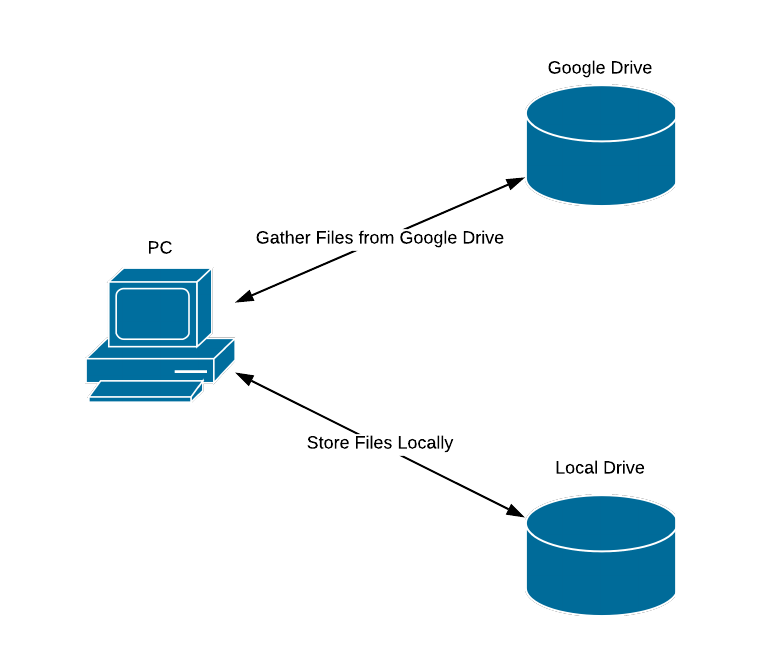
#### 2.3.1 Working on a group project

Users contributing to a group project that is owned by another group member may desire the ability to backup group files to their own devices so that if the owner of the file revokes access or deletes the file the data is still accessible by the user.

#### 2.3.2 Distrust of third party storage

Users concerned with files in Google Drive being the only extant copy may want to back up the contents of their Google Drive to a local device so that in the event that they lose access to their Google account their data is not lost.

### 2.4. User Diagram



## Behavioral Requirements

### 3.1. System Inputs and Outputs

#### 3.1.1 Inputs

[IR 1] Configuration

[IR 1.1] FileType Filters

The user may specify what file types they would like to backup.

[IR 1.2] Scheduling

If the user would like to schedule automated backups, they must specify how frequently the backups should occur.

[IR 1.3] Logging Directory

The user must specify what directory the logfiles will be written to.

[IR 1.4] Rotation

If the user wants backups to be automatically removed after a certain period of time, they must specify it.

[IR 1.5] Backup Directory

The user must specify what directory the backups will be written to.

[IR 2] User Credentials

The user must provide their Google username and password for the API to return an access token.

#### 3.1.2 Outputs

[OR 1] File Backups

[OR 1.1] File Downloads

The primary output of Drive2Local is a collection of files downloaded from the Google Drive of the user. These files will be stored in a time stamped directory so the user is able to tell when a backup was made. The type of files backed up will match those specified in the FileType Filters set by the user.

[OR 1.2] Backup Automation

If automatic backups are enabled, these backups will occur at a frequency set by the user.

[OR 1.3] Backup Rotation

If rotation is enabled, old backups will be automatically deleted after a number of backups specified by the user are accrued.

[OR 2] Logging Information

[OR 2.1] Metadata Tracking

The logs will contain information retrieved from file metadata including the time and date of changes to files and which user last contributed to the files.

[OR 2.2] Backup Success

The logs will contain information pertaining to the success of the backup, as well as any potential errors the program may have generated.

[OR 2.3] File Diffs

The logs will contain information about what changes were made to files between backups.

### 3.2. Detailed Output Behavior

Drive2Local will use the configuration settings provided by the user to download files (limited to a specific subset of filetypes if specified by the user) that a user’s Google account has permission to read and store them in a directory on the user’s local device in a time stamped folder. If the user has automatic backups and file rotation enabled, backups will occur automatically at a rate set by the user and be deleted after a specific number (also set by the user) are acquired.

## Developmental Quality Attributes

[DQA 1] Documentation

[DQA 1.1] Code Commenting

Each block of code will have an associated comment describing what it does and how it does it.

[DQA 1.2] Project Documentation

The Project Plan, SRS, SDS, and other related documents will be updated to reflect any changes made to the project, either in terms of project management or software architecture.

[DQA 2] Ease of change

[DQA 2.1] Source Code Accessibility

All members of the development team will have write access to the source code and be able to push changes.

[DQA 2.2] Version Control

Version control will be utilized to prevent large changes to the codebase substantially altering the overall project at any point.

[DQA 3] Modularity

Each major function of the program (file downloading, object definitions, local storage handling, config handling, etc.) will be contained within its own module.

[DQA 4] Consistency

Code will be written and designed in a consistent manner. Naming conventions and indentation will be the same across code written by any member of the development team.

## Fundamental Assumptions

We assume the user is running Drive2Local on a version of MacOS 10.12 or greater or Ubuntu Server 16.04 or later. The software will likely run on operating systems outside of this scope, but these are the ones we officially support.

Python 3.7.0 or later is required.

The python modules apiclient, google-api-python-client, google-auth-httplib2, google-auth-oauthlib, oauthclient2, and python-crontab are required.

It is also assumed the user has access to the internet while running the software and that they have a Google account with files stored in their Google Drive.

# III. Software Design Specification (SDS)

Revision History:

|  |  |  |
| --- | --- | --- |
| Date | Author | Description |
| 02/20/19 | Chen, Shea-Bianco, Zhang | Initial draft |
| 02/21/19 | Chen, Shea-Bianco, Wilson, Zhang | Edit and complete initial draft. |
| 03/04/19 | Wilson | Update models and functions. |
| 03/05/19 | Wilson | Update module interface specification |
| 03/07/19 | Wilson | Update Logging module |
| 03/10/19 | Wilson | Update SDS |
| 3/11/2019 | Caraballo, Wilson | Updated SDS |

## Software Architecture

### 1.1 Modules

**User Interface(Drive2LocalUIMain.py)**

The purpose of this module is to create a graphical user interface for the user to choose the settings they would like to use during backup. It also offers a way for the user to backup their files right away. This is a pretty basic UI, which uses PyQt5 library to help set up the GUI and it allows us to do all the commands that we need.

Imported statements:

“Imported sys”  
 “from PyQt5 import QtCore, QtGui, QtWidgets”

“From Drive2LocalUISettings import Ui\_Setting”

“Import Drive2Local”

Class Object:

Ui\_Main(Object)

Methods:

openSetting()

StartMainFunction()

setupUI()

retranslateUI()

**User Interface Settings(Drive2LocalUISetting.py)**

The purpose of this file is to help with the settings file in the GUI. When the user clicks on the “Settings” button, it will direct them to another page which will allow the user to change different aspect of the program. On this page this gives the user the option to change the directory in which they want to put their backed-up files from Google Drive. As well as change the location of the log file that will show them what download at what time. Also some other cool aspects this allows the user to change is what type of files they want download from their Drive. Also we allow the user to set up recurring back-up times on a daily basis. This Drive2LocalUISetting.py creates the page in the GUI and links the different buttons to perform these tasks.

Import statements:

“import sys”

“from PyQt5 import QtCore, QtGui, QtWidgets”

“from PyQt5.QtWidgets import QFileDialog”

“import Drive2LocalConfig as config”

Class Object:

Ui\_Setting(object)

Class Methods:

open\_backup\_dir(self)

open\_log\_dir(self)

setupUI(self, Setting)

accept(self)

retranslateUI(self, Setting)

#### **Main(Drive2Local.py)**

The purpose of this module is to connect all the modules together in a central location where the program can be run from. Drive2Local.py is used when the “Backup” button is triggered in the GUI and this file is considered the driver on the project.

Import statements:

“import Drive2LocalAPIAccess, Drive2LocalHandleLocal, Drive2LocalLogging”

“from Drive2LocalConfig import \*”

Methods

isOwned(file)

Takes a file object and returns true if it’s owned by the user of the drive and false otherwise.

isTrashed(file)

Takes a file object and returns true if the file exists in the trash and false otherwise.

isFilteredExtension(file)

Checks whether or not the file extension is in the filtered list.

isGoogleFile(file)

Checks whether or not the file is a native Google file.

main()

Makes calls to the API Access, Config, and Handle Local modules in order to begin backing up the user’s files with the right settings.

#### **API Access(Drive2LocalAPIAccess.py)**

The purpose of this module is to interact with the Google Drive API to send and retrieve data.

Methods

getDrive

Takes no arguments, but rather creates a JSON file to keep track of user login token and authorizes read-only access to the user’s entire Google Drive.

getFiles

Takes an instance of a user’s Google Drive as an argument. Returns a list of all the file objects stored in the Drive. Each object contains all the metadata of each file.

getFilesMin

Takes an instance of a user’s Google Drive as an argument. Returns a list of all the file objects stored in the Drive. Each object contains only the name, id, and MIME type.

getFileMetadata

Takes a drive instance and a file id as arguments and returns a file object populated with all the metadata for that file.

downloadFile

Takes a drive instance, file id, and file name as arguments. Then attempts to download the file.

exportFile

Takes a drive instance, file id, MIME type, and file name as arguments. Then tries to export the file from the user’s drive formatted as the MIME type given.

#### **Configuration(Drive2LocalConfig.py)**

The purpose of this module is to configure various user settings. It contains no methods or functions, but instead it contains variables that represent the different user settings.

Settings

filetype\_filter

Whether or not to filter files based on file type.

filetypes

The list of file types that will be filtered out of the drive.

owner\_filter

Whether or not backups are limited to the files owned by the user.

trash\_filter

Whether or not to backup files that have been trashed.

automatic\_backups

Whether or not to automatically create backups.

backup\_frequency

The frequency of scheduled backups measured in days.

rotation\_on

Whether or not it automatically deletes old backups.

rotation\_num

The frequency of automatic file deletion, measured in how many backups to keep of each file.

backup\_root

The target directory for file downloads to write to.

#### **Local File Handling(Drive2LocalHandleLocal.py)**

The purpose of this module is to handle the local filesystem, including creation and deletion of files.

Methods

buildDir

Builds a local file structure for storing downloaded backups.

writeFile

Takes an instance of a user’s Google Drive, file path, and a file object as arguments. Then calls the API Access module using the arguments to export the file to the correct location.

writeGoogleFile

Takes an instance of a user’s Google Drive, file path, and a file object as arguments. Then calls the API Access module using the arguments to export the native Google file to the correct location.

compressDir

Takes a path to a directory as an argument and compress the directory into a zip file.

rotateBackups

Deletes old backups until there are only a user specified number of most recent backups in the backup directory.

scheduleBackups

Sets a scheduled task for the system to rerun Drive2Local and generate new backups automatically.

#### **Logging(Drive2LocalLogging.py)**

The purpose of this module is to format and setup a logger that will write logs to a log file.

Methods

infoLog

Takes a message and records within the log file a message, name of file, and a timestamp for when the error occurred.

debugLog

Takes a message and records within the log file a message, name of file, and a timestamp for when the error occurred.

errorLog

Takes a message and records within the log file an error, message, name of file, and a timestamp for when the error occurred.

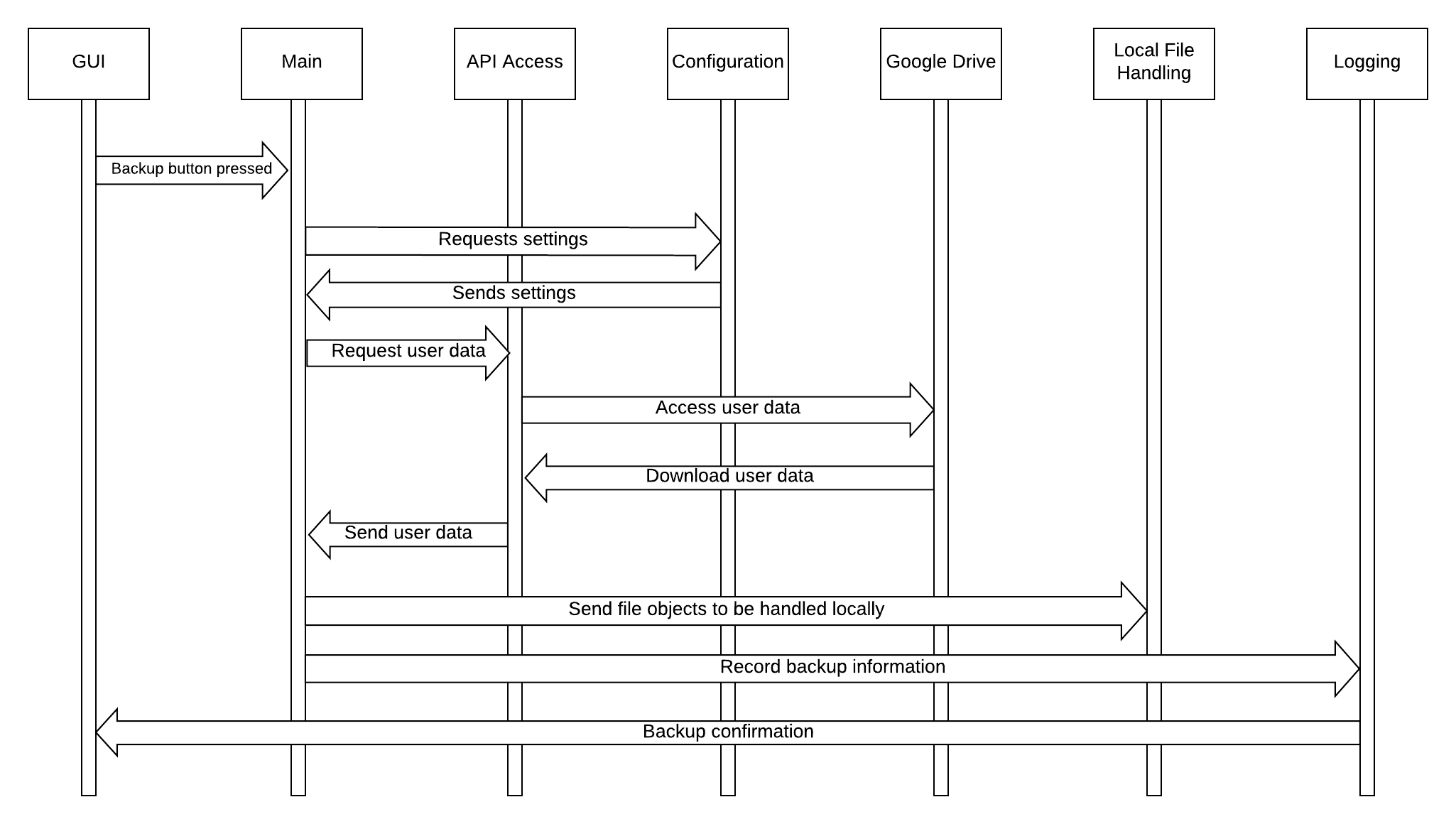
### 1.2 Module Interaction

The main module will interact with the API Access, Local File Handling, and Configuration modules to start the process of backing up the user’s Google Drive. The API Access module uses login credentials provided by the user during their initial login to their Google account to access their Google Drive. The Config module stores the user settings that will determine the functionality to be used during the backup process. The Handle Local module will make calls to the API Access module to setup the backup file structure and write files to the user’s local machine. Throughout this program the Logging module will be called to record when and what files are being downloaded.

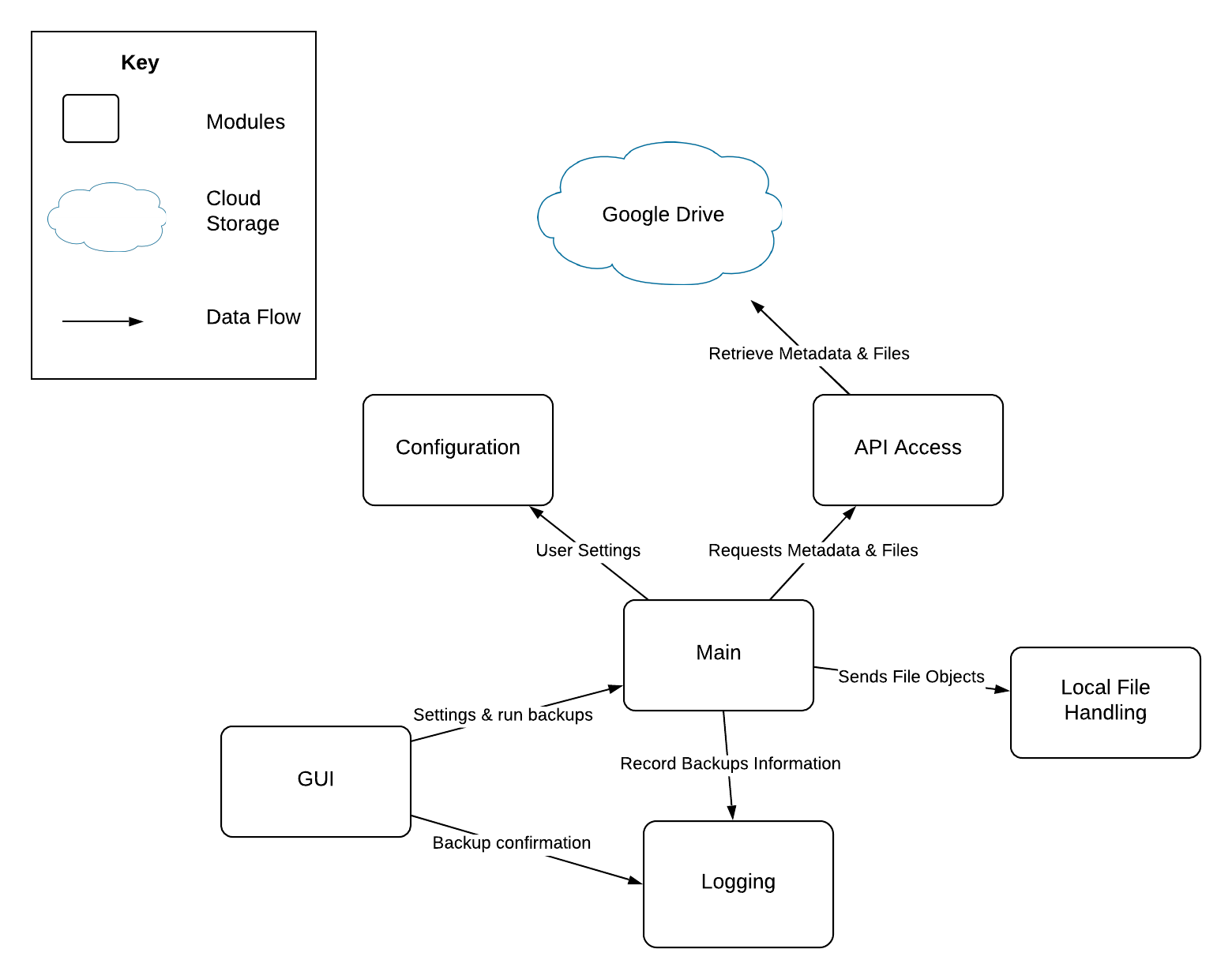
Static and dynamic models detailing the inter-module interactions can be found on the following pages.

### 1.3 Design Rationale

We chose to break the system into these components because this decomposition made the most sense to the entire team. The modules listed above have low enough coupling that each module could be worked on independently, while possessing strong cohesion in terms of scope.



Dynamic Model



Static Model

## Module Interface Specification

### 2.1 Main to Configuration

Client: Main

Server: Configuration

The Main module gathers user settings from the Configuration module so that it is able configure backups accordingly for the user’s Google Drive.

### 2.2 Main to Logging

Client: Main

Server: Logging

After the Main module makes any changes to the system, it will call the Logging module and pass in data to be written to log files.

2.3 Logging to GUI

Client: Logging

Server: GUI

After the Main module makes any changes to the system, it will call the Logging module and pass in data to be written to log files.

### 2.4 Main to API Access

Client: Main

Server: API Access

The Main module uses the API access module to access file objects that are stored within the user’s Google Drive.

### 2.5 API Access to Google Drive

Client: API Access

Server: Google Drive

While Google Drive is not a module developed as part of Drive2Local, it is important to note the interactions between it and the API Access module. The API Access module acts as a client in this interaction, using the API specified by Google to request files and file metadata from Google Drive by providing a key generated using the user’s username and password.

### 2.6 Main to Local File Handling

Client: Local File Handling

Server: API Access

The Main module will use the Local File Handling module to create the correct file structure locally on the user’s computer.

2.7 GUI to Main

Client: GUI

Server: Main

The Main module will use the Local File Handling module to create the correct file structure locally on the user’s computer.

# 

# IV. Quality Assurance Plan

# Revision History:

|  |  |  |
| --- | --- | --- |
| Date | Author | Description |
| 02/21/19 | Chen, Shea-Bianco, Zhang | Initial draft |

## **Reviews**

Include these three types of reviews:

* Peer review:
  + The evaluation of an artifact or its performance by peers in order to maintain or enhance the quality or performance of the artifact.
* Formal Review:
  + A structured examination of an artifact by an assigned formal review team.
* Milestone Review:
  + Milestone Reviews are mandatory and ensure that the work required in the current state or increment is complete.
  + Project or increment is ready to enter the next state or increment.
  + Milestone 0 - Design and required analysis complete
  + Milestone 1 - Required function implementation
  + Milestone 2 - Testing all the functions and bug fixing
  + Milestone 3 - Complete programer’s documentation and developer logs
  + Milestone 4 - Project complete

## Interviews with Potential Users

* 1st interview: Ricky Smith, 4th year undergraduate student at the University of Oregon, Business Major. The interview took place on March 7th, 2019. During the interview, we were able to explain how Drive2Local works, showed him how it works, and explained why we think this is a very applicable program. Then I was able to ask Ricky some questions to see what his thoughts are on it. The questions I was able to ask are below:
  + Q: “How often do you use Google Drive and what for?”
  + Q: “How often do you use Google Drive to share documents with other people for group work?
  + Q: “Do you think Drive2Local is something you, your group, or anybody else would use?”
  + Q: “Is there anything that you would possibly change or are there any features that you would recommend to make it more applicable to yourself or possibly anyone you know?”
  + Q: “Do you think the program is pretty straight forward/easy to use?”
  + Q: “Is there anything about the project you just don’t like?”
* The interview with Ricky Smith was very helpful and it was great to get some advice from an outside source. Based off the interview we were able to learn a lot, first of all Ricky Smith is a frequent Google Drive user which helped us out and understand that Drive2Local can possibly be applicable to him. Ricky also expanded on the fact that he uses Google Drive quite a bit for group work specifically because it allows him to edit documents in real time with his group. Most of the work he does in his classes is group work so he is always using Google Drive.

There was pretty positive feedback about Drive2Local that Ricky gave us. At the time the GUI wasn’t fully developed. However I was able to show him through terminal how the program does work. He gave some really good feedback and had some great questions. His first concern was how the GUI would look and he wanted to make sure it wouldn’t be so complicated since the whole concept of the project might be pretty difficult to understand for someone who isn't so tech-savvy. This led the group to consider the complexity and abstractness of the GUI. We figured the best design for the UI is to be as simple as possible. We have a “Backup” button which allows us to download the Google Drive files. As well as a “Settings” button that allows us to change the location of where our files will go on our local computer, certain type of files that we will download, and set recurring backup times throughout the day.

Some other important aspects that he really made us consider is that this project is very useful to a lot of people and it can be used in a lot of different situations. However there are some things that could taken into account. He suggested that we eventually support single file back ups since there is more of a need to download a single file instead of the whole drive. Which is a really good point however, that is something that can be implemented later in time.

**Testing**

* Goals:
  + This Quality Assurance Plan (QA) details the overall approach to quality assurance activities for Drive2Local. This QA Plan documents how the project defines, implements and assures quality during the software development process.
  + This QA Plan is a communication vehicle for the entire project team, including the project manager, architect, developers, test & Integrator, configuration control, other project teams, and users.
* Test Kind:
  + Unit Testing:
    - A level of the software testing process where individual units of a software are tested. The purpose is to validate that each unit of the software performs as designed.
  + Integration Testing:
    - A level of the software testing process where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units.
  + System Testing:
    - A level of the software testing process where a complete, integrated system is tested. The purpose of this test is to evaluate the system’s compliance with the specified requirements.
  + Acceptance Testing:
    - A level of the software testing process where a system is tested for acceptability. The purpose of this test is to evaluate the system’s compliance with the business requirements and assess whether it is acceptable for delivery.
* Test Actually Run:
  + WIP

# V. Software Documentation

# Revision History:

|  |  |  |
| --- | --- | --- |
| Date | Author | Description |
|  |  |  |

This software for the use of Google documents for customers, users, and so on with the system supporting the local backup function. This program belongs to a third party plug-in. After installation, users using this feature need to have their own Google account to log in the software.

After logging in, you can enter an address of a Google document that you want to download locally. And can be selected through the filter function to think about the type of files downloaded or a part of the file. When downloaded locally, the user can edit the file without the network.

## 

## 

## 

# VI. Developer Logs

WIP

1. https://www.recode.net/2016/7/29/12312086/millenials-microsoft-word-google-docs-collaboration-study [↑](#footnote-ref-0)